A PERFORMANCE EVALUATION OF AN ALPHA EV7 PROCESSING NODE

Darren J. Kerbyson
Adolfy Hoisie
Scott Pakin
Fabrizio Petrini
Harvey J. Wasserman

LOS ALAMOS NATIONAL LABORATORY (LANL), CCS-3 MODELING, ALGORITHMS AND INFORMATICS GROUP, PERFORMANCE AND ARCHITECTURES LABORATORY, LOS ALAMOS, NM 87545, USA

Abstract

In this paper we detail the performance of a new AlphaServer node containing 16 Alpha EV7 CPUs. The EV7 processor is based on the EV68 processor core that is used in terascale systems at Los Alamos National Laboratory and the Pittsburgh Supercomputing Center. The EV68 processor core is supplemented with six-way router circuitry that forms connections from the processor internals to four neighboring CPUs in a two-dimensional torus, to a I/O controller and to local memory. The performance evaluation presented in this paper considers memory hierarchy, intra-node MPI communication, and also the performance of a number of complete applications. The measurements are compared with those taken on existing AlphaServer machines. It is clear from our analysis that the superior application performance of the EV7 relative to a similar-speed EV68 is attributable to its excellent main memory bandwidth – over 4 GB/s.

Key Words: Performance, analysis, high performance computing, memory performance, communication performance, application performance

The International Journal of High Performance Computing Applications, Volume 18, No. 2, Summer 2004, pp. 199–209
DOI: 10.1177/1094342004039808

1 Introduction

This paper details a performance evaluation of a state-of-the-art AlphaServer node. This node is a prototype of the latest generation AlphaServer systems, which are designed to scale up to 128 processors per node. The 21364 processor, code-named EV7, is the latest in the Alpha processor line. The EV7 design emphasizes high memory performance in order to overcome some of the performance issues associated with the increasing gap between processor and memory speeds. The performance of a 16-processor node is examined here in terms of memory performance, intra-node MPI communication performance, and also with a number of complete applications. The performance reported through this work is taken from a pre-production node running at a clock rate of 1.2 GHz. The production systems available from HP at the start of 2003, designated the AlphaServer GS1280, actually run at a clock-rate of 1.15 GHz.

The most significant changes relative to the current HP AlphaServer ES40 (Cvetanovic and Kessler, 2000) and ES45 (Compaq, 2002) nodes include: an upgrade of the CPU from the EV68 to the EV7, PCI-X I/O slots, and a ccNUMA memory architecture (Krewell, 2002). The EV7 CPU is based on a slightly modified EV68 core but augments it with two on-chip Direct Rambus (RDRAM) memory controllers, an on-chip 1.75 MB L2 cache, and a six-port router. In both the EV68 and the EV7, a maximum of two floating-point operations can be executed each cycle, so a 1.2 GHz CPU has a peak theoretical processing rate of 2.4 GFLOPS. The EV7 improvements to the EV68 core include the ability to accommodate 16 concurrent outstanding cache misses, versus eight in the original core.

L1 and L2 cache latencies are the same in the EV68 and the EV7: a two-cycle latency to the L1 cache and a 12-cycle latency to the L2 cache. The EV7 L2 cache is seven-way set associative and can transfer data to the CPU at 16 bytes per cycle (a peak of 19.2 GB/s at 1.2 GHz). Note that the EV68 L2 cache was much larger (up to 16 MB) but was off-chip and had a maximum transfer rate to the CPU of only 5.3 GB/s. The two on-chip RDRAM memory controllers of the EV7 support a maximum memory-to-L2 transfer rate of 12 GB/s, versus the EV68's maximum of only 2.6 GB/s.

The EV7 chip also includes a router with a total of six connections (Mukherjee et al., 2002). Four connections go to neighboring processors arranged within a node as a two-dimensional (2D) torus topology. These are capable of running at a peak of 6.4 GB/s each. One connection links directly to external I/O, and the remaining connection integrates the local processor resources, such as the core and the two memory controllers. This operates at the same speed as the processor core and has a low pin-to-pin latency. In comparison, the ASIC-based SGI Spider router runs at 100 MHz with a much larger latency (Galles,

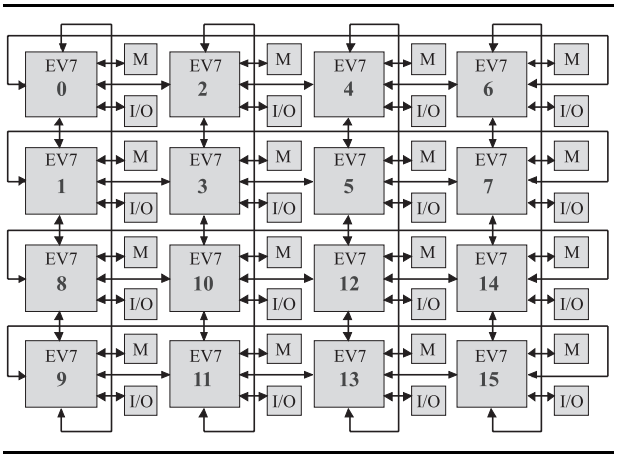

Fig. 1 A 16 processor EV7 node indicating processor ID ordering.

1987). The arrangement of processors within a single 16-CPU node is shown in Figure 1.

A node is composed of between 1 and 128 EV7 CPUs interconnected via the on-chip routers. The resulting system is a ccNUMA (cache coherent, non-uniform memory architecture) design in which any CPU can address all of the memory within the node but the memory access time differs depending on the proximity to the data. Each reference to non-local memory includes a local-memory latency, an overhead to get in and out of the network, and a cost per hop which is mostly wire and router delay.

A node is physically constructed from two-CPU processor boards. This is reflected in the processor ID ordering indicated in Figure 1; there are eight pairs of processors in the 16-CPU node with each board housing a vertical pair of processors in the 2D torus network. The interconnection between adjacent processors is thus physically different for processors on the same board versus different boards. Furthermore, connections that wrap around the torus are physically different from those that do not. The differences in the physical interconnection are reflected to some extent to the remote memory access latency and also to the interprocessor communication latency (Sections 2 and 3).

The AlphaServer node that we analyzed contained 16 EV7 processors. Three sets of tests were used to analyze its performance:

(1) memory hierarchy performance micro-benchmarks;
(2) intra-node MPI communication kernels; and
(3) several complete applications.

The performance of the memory hierarchy is detailed in Section 2, the performance of the intra-node MPI communication is detailed in Section 3, and the performances of several applications are detailed in Section 4. A comparison is made in Section 5 between the performance measured on this node to that measured on existing AlphaServer machines.

2 Memory-Hierarchy Performance

The performance of the memory hierarchy is analyzed here in terms of the latency of the various levels of the hierarchy and the bandwidth observed when transferring data to local and remote memory.

2.1 MEMORY LATENCY

Memory latency was measured by reading from a vector in which successive reads are from elements a cache-line length apart. This guarantees that each memory access will exhibit a latency cost to memory as no spatial cache-reuse will be possible. By increasing the size of the vector, the latency to different parts of the memory hierarchy can be observed. After repeated runs, a short vector may fit into one of the caches and therefore require no access to further parts of the memory hierarchy while a long vector may overflow all of the caches and expose main-memory latency. In addition, the memory vector can be placed on a pre-determined PE (Processing Element, meaning CPU + memory) within the node and read from a different predetermined PE. We can thereby separately measure local and remote cache misses.

Figure 2 shows the measured memory latency from remote memory when the read operation is performed by processor 12. For each of the other processors, the processor distance in processor-hops is shown. Several plateaus in the curves can be seen: a vector of size up to 64 KB can be held within the local L1 cache, a vector of size up to 1.75 MB can be held within the local L2 cache, and vector of sizes greater than 1.75 MB must reside in the main memory. The main memory latency varies widely due to the distance between processors.

A summary of the memory latency measured for the 64 Mbytes vector size is shown in Figure 3. The processor-hop distance from processor 12 for this experiment is also shown in a small box on each processor. The memory latency shown for processor 12 is the latency to its own local memory. The processor ID ordering corresponds to that shown in Figure 1.

The latency to the remote memory increases as the distance (in processor-hops) increases. It also depends on the route taken between the processors. That is, a one-hop access to a processor located on the same processor board will observe a lower latency than a one-hop access to a processor on a different board. The latency to local memory was measured at 106 cycles. Each reference to non-local memory pays this same local-memory penalty plus an overhead to get in and out of the network of approximately

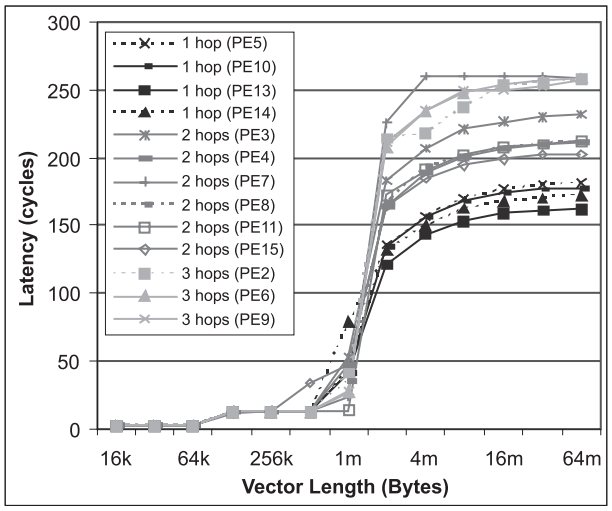

Fig. 2 Memory latency from remote processors to processor 12.

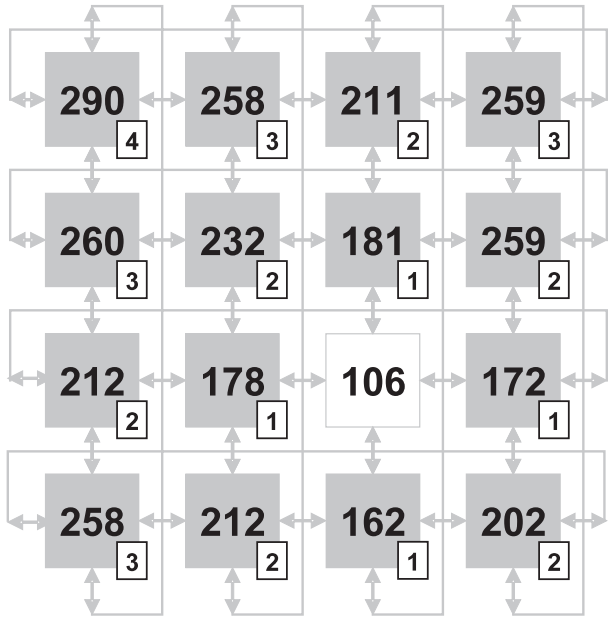

Fig. 3 Remote memory latencies to processor 12 (clock cycles).

25 cycles, plus an overhead per processor-hop of approximately 45 cycles, which is mostly wire and router delay.

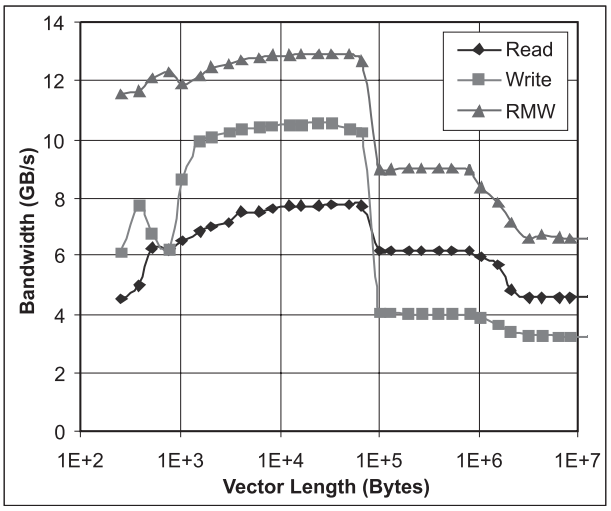

Fig. 4 Achievable peak memory bandwidths (read, write, Read-Modify-Write).

The total delay to read the memory on a processor that is one hop away is therefore about 176 cycles. Each additional hop adds, on average, a further 37 cycles of delay.

The *worst-case* latency on the 16-CPU node is 290 cycles (242 ns at 1.2 GHz), which is just under three times the best case of 106 cycles (88 ns). If the same performance characteristics are assumed for a 64-CPU node, the worst-case latency would be roughly 356 ns (3.5 times worse than the best case). However, the *average* memory latency across such a node would approximately be 230 ns, which compares favorably with the (uniform) memory latency of 170 ns on an Alpha EV68 ES45 four-processor node. Smaller nodes will have smaller average latencies (e.g., about 200 ns on a 32-CPU node, and 170 ns on a 16-CPU node).

2.2 MEMORY BANDWIDTH

The memory bandwidth was measured using Cachebench (Mucci and London, 1998). This was used to measure the peak achievable memory bandwidth on a single processor for read, write, read-modify-write, memset, and memcpy. The results of using Cachebench are shown in Figures 4 and 5 using the double data-type in all cases. All results are measured for a vector of varying size.

In addition, a vector of size larger than the local processor memory was allocated and a memory read operation performed. This test results in data being placed both in a

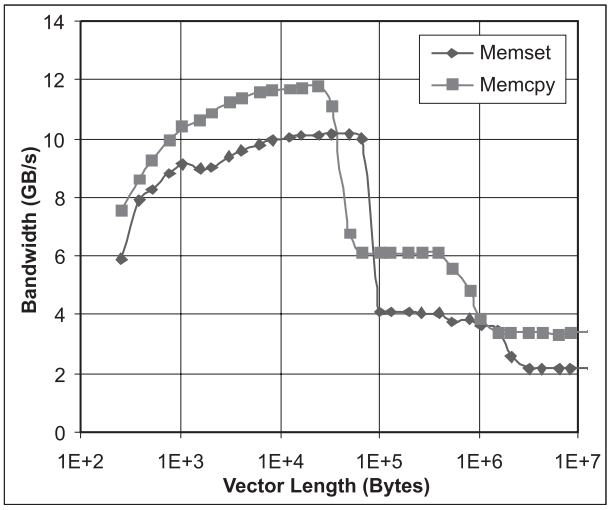

Fig. 5 Achievable peak memory bandwidths (memset, memcpy).

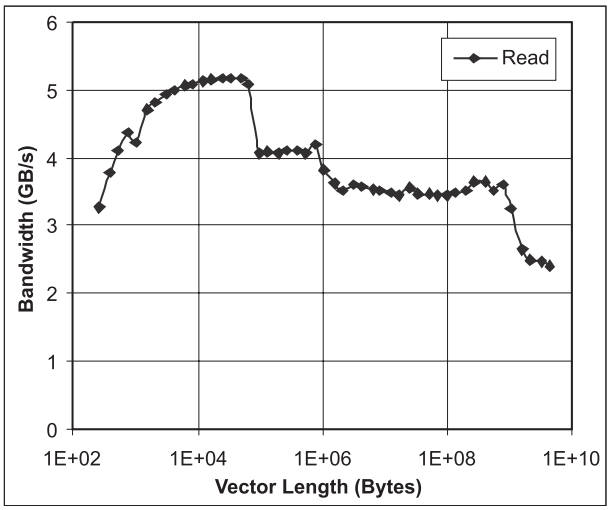

Fig. 6 Achievable peak memory bandwidth (read) illustrating remote memory access on large vector sizes.

processor's local memory and also in a remote memory. The bandwidth obtained is shown in Figure 6. Note that this test was performed only on a slower EV7 node running at 800 MHz but is included here for comparison. The first portion of the curve in Figure 6 up to a vector of length 10 MB corresponds to the read curve in Figure 4 (noting the difference in the clock speeds). The bandwidth to remote memory can be seen for a vector of length greater than 2 GB.

Table 1
Memory performance summary.

	Peak bandwidth (GB/s)	Latency (cycles)
L1 cache (64 KB)	7.77	2
L2 cache (1.75 MB)	6.20	12
Main memory (2 GB)	4.60	106
Remote memory (16-CPU node)	~3.60	162–290

It is expected that the performance of memset corresponds to that of the write performance and that the performance of memcpy corresponds to that of the read-modify-write performance. As can be seen from Figures 5 and 6, they are indeed similar for L1 cache performance but both underperform on main memory indicating a better implementation may be possible.

A second test measured the available memory bandwidth on a single processor while a number of other processors within the node continuously read from their local memory to measure the effective bandwidth in the presence of contention. The impact of having other processors perform background memory reads did *not* have an impact on the given processor's memory performance. This is unlike many of the existing smaller SMP nodes, such as the AlphaServer ES45, which can be affected by a reduction in bandwidth of a factor of 2 due to memory bus contention. The measured data are not included here.

The memory bandwidth observed for the EV7 node is very good and is summarized in Table 1. Note that the size of each level of the memory hierarchy can be seen in Figure 6 by the three plateaus in the memory bandwidth.

The results show that there is less than a factor of 2 bandwidth reduction between the L1 cache and main memory, thus illustrating a major strength of the EV7 processor.

3 Intra-Node Communication Performance

The intra-node communication performance was measured using a number of MPI-based tests:

- Ping-pong message performance between two adjacent processors in a node. This was measured for both unidirectional and bidirectional message traffic, recording both message latency and bandwidth.
- Message latency and bandwidth between a single processor and all other processors in a node to indicate the performance between non-adjacent processors.
- Hot-spot communication performance – the achieved communication bandwidth when more than one processor communicates to a single processor.

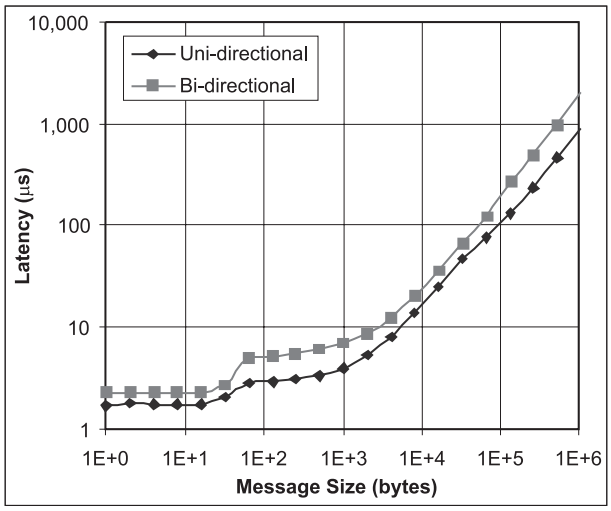

Fig. 7 MPI message latency between two adjacent PEs.

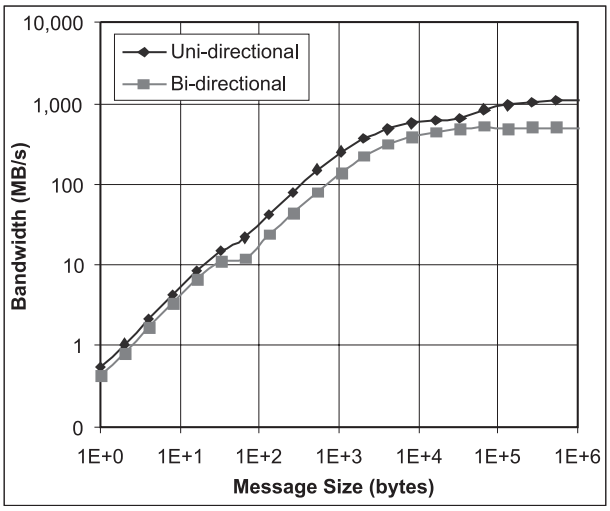

Fig. 8 MPI ping-pong message bandwidth between two adjacent PEs.

- Barrier performance – latency for an MPI barrier.
- Broadcast performance – achieved bandwidth for an MPI broadcast.

3.1 MPI COMMUNICATION PERFORMANCE

The performance of both unidirectional and bidirectional MPI communication between two adjacent processors within a node using a ping-pong test is shown in Figures 7 and 8. Figure 7 shows the duration (latency) of the communication and Figure 8 shows the achieved bandwidth for messages of size between 1 and 1,000,000 bytes.

The achieved latency of a small message was 1.7 μs in a unidirectional communication and 2.2 μs for a bidirectional communication. The ping-pong bandwidth on a message of size 1 MB was 1.08 GB/s for a unidirectional communication and 485 MB/s for bidirectional communication. Note that the bidirectional bandwidth is quoted for the achieved bandwidth for each direction in the communication. The bidirectional bandwidth in each direction is just under half of the unidirectional bandwidth.

3.2 POINT-TO-POINT COMMUNICATION PERFORMANCE WITHIN A NODE

The performance of a unidirectional communication using a ping-pong test was recorded for all processors within a node when communicating with processor 0. The latency obtained for a 0-byte message and the bandwidth achieved on a message of size 1 MB are shown in Figures 9 and 10, respectively. The processor-hop distance from processor 0 for this experiment is also shown in a small box on each processor.

Both figures show a 4×4 processor map of the PEs in a node. The latency increases as the distance in processor-hops increases. The bandwidth between PE 0 and any other PE is approximately a constant at 1.08 GB/s. Processors on the same board have a slightly lower latency than those that are not. Each vertical pair of processors resides on the same board, thus the latency between PE 0 (top-left) and PE 1 (second from top on left) is smaller than the latency between PE 0 and PE 2 (top, second from left). PEs within a node are connected in a 2D torus topology and thus the PE (lower-right) is only two hops distant from PE 0.

3.3 HOT-SPOT COMMUNICATION PERFORMANCE

The achieved bandwidth performance under the hot-spot communication traffic is shown in Figure 11. The hot-spot communication tests the situation when one or more processors simultaneously communicates to a single processor in a repetitive mode. In the case shown in Figure 11 one or more processors repetitively sent a message of size 256 KB to processor 0.

The achievable bandwidth on this test actually increases slightly as more processors perform the simultaneous

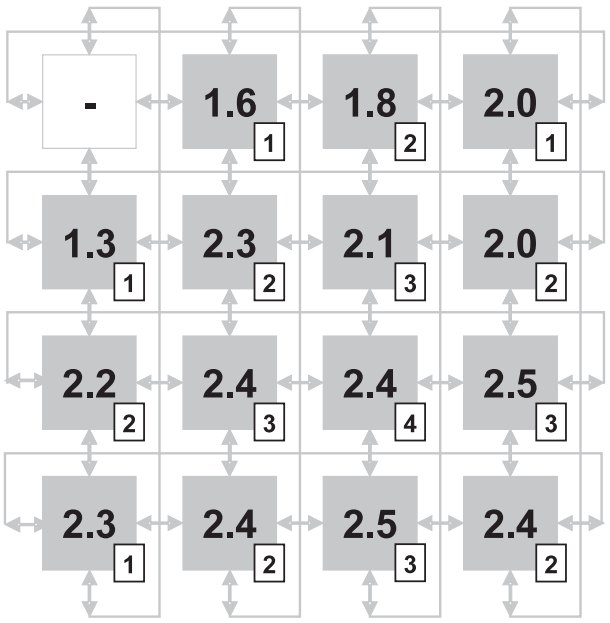

Fig. 9 MPI latency (μs).

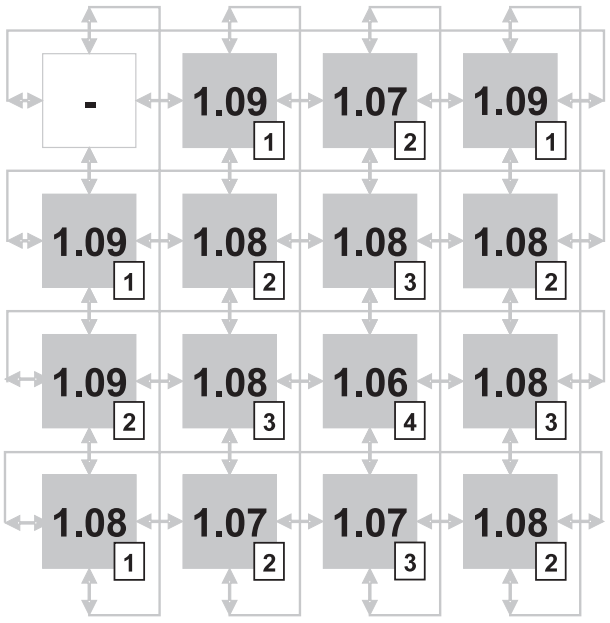

Fig. 10 MPI bandwidth (GB/s).

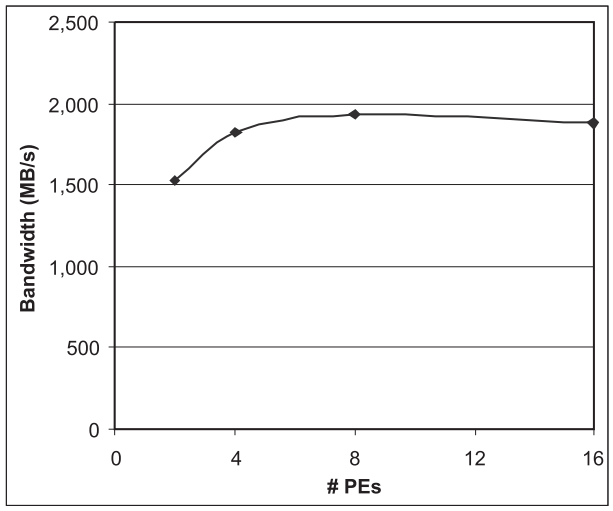

Fig. 11 Achievable bandwidth under the hot-spot traffic.

communication and approaches a maximum of just over 1.9 GB/s. This indicates that the available bandwidth to a single processor exceeds that which can be utilized by a single pair of processors.

3.4 MPI BARRIER LATENCY AND BROADCAST BANDWIDTH

The performance of the MPI barrier is shown in Figure 12. The barrier takes 11.2 μs across all processors in the 16-CPU node. It is interesting to note that this barrier time is actually larger than on a Quadrics QsNet network which takes 7 μs for an inter-node barrier on 512 nodes (Petrini et al., 2002). The performance of the MPI broadcast is shown in Figure 13. The achievable bandwidth decreases as more processors are involved in the broadcast. This is due to the broadcast operation relying on point-to-point messages propagating through the 2D torus topology. In contrast, the Quadrics QsNet network uses additional hardware support to improve its barrier and broadcast performance. The bandwidth decreases from 1.01 GB/s on two processors down to 300 MB/s when using all 16 processors in the EV7 node.

4 Application Performance

The performance of several applications of interest to LANL was measured on the EV7 AlphaServer node. Performance is detailed here for SAGE, MCNP, and

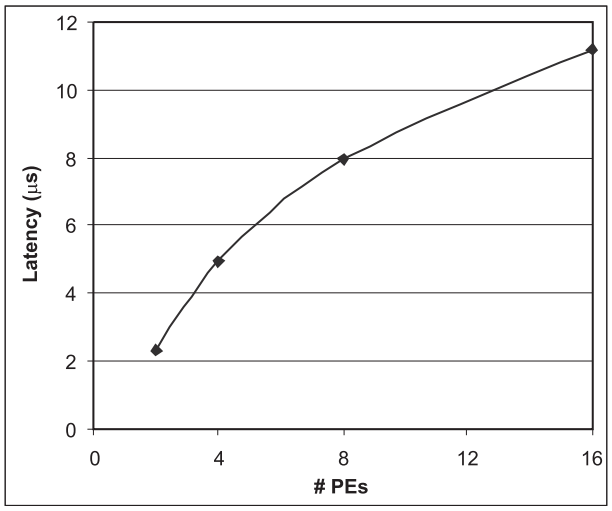

Fig. 12 MPI barrier performance.

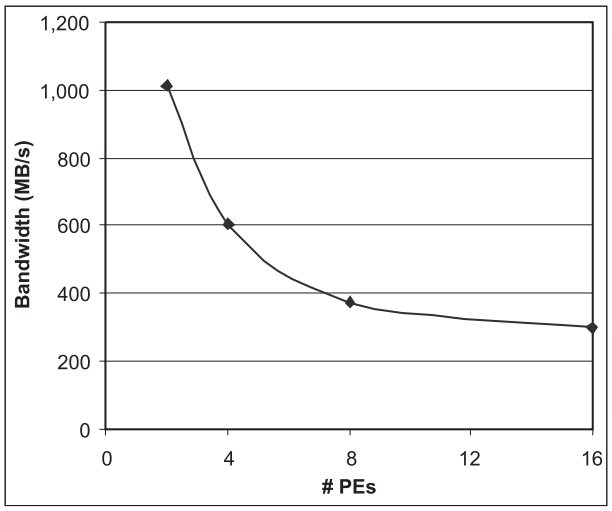

Fig. 13 MPI broadcast performance.

SWEEP3D. SAGE is a multi-dimensional multi-material hydrodynamics code with adaptive mesh refinement. A description of SAGE is given in (Kerbyson, 2001). MCNP is a general purpose Monte Carlo N-Particle code that can be used for neutron, photon, electron, or coupled transport (McKinney, 1994). SWEEP3D is a time-independent, Cartesian-grid, single-group, discrete ordinates deterministic

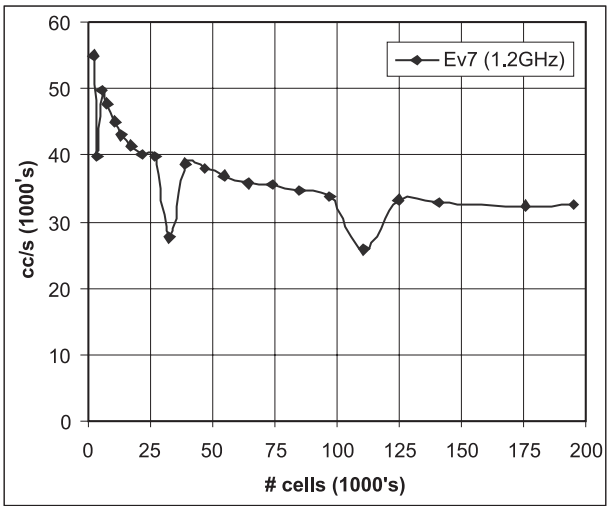

Fig. 14 Sequential performance of SAGE when varying the spatial grid size.

particle transport code (Koch et al., 1992). The performance of each application was examined in a number of different configurations as described below.

4.1 SAGE

The performance of SAGE was examined in two different studies. The first considers a sequential test whilst varying the size of the spatial grid used in terms of the number of cells processed in a single iteration of the code. This is to examine the impact of the memory hierarchy as it is possible for small spatial grids to be L2 cache resident whereas larger grids are not. The second study considers a single spatial grid size whilst varying the number of processors used in a weak-scaling study (i.e., keeping the number of cells per processor at a constant). Both studies are described below.

(i) SAGE – Spatial grid scaling on a single processor.

This is a sequential test of SAGE whilst scaling the number of cells in the spatial grid. The number of cells was varied from 14^3 to 58^3. Note that SAGE uses a three-dimensional (3D) spatial cube by default; hence, the number of cells was varied as a cubic power. The result of this scaling is shown in Figure 14 using the number of cells that can be processed in one second (cc/s) – a rate-based metric. Ideally this should be a constant for all problem sizes.

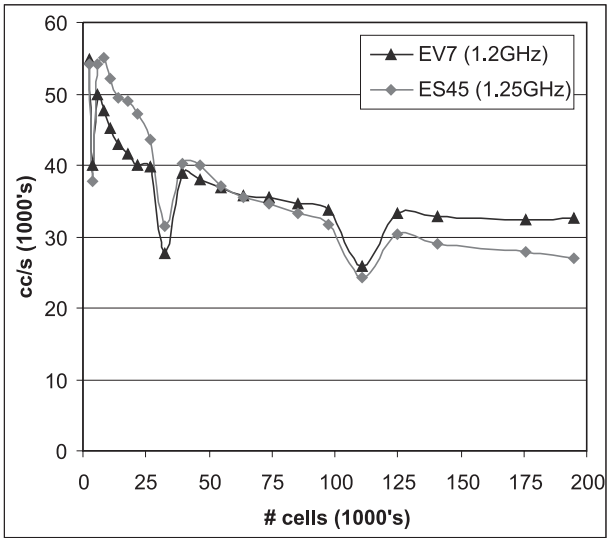

Fig. 15 Comparison of SAGE performance on the EV7 (1.2 GHz) and the EV68 (1.25 GHz).

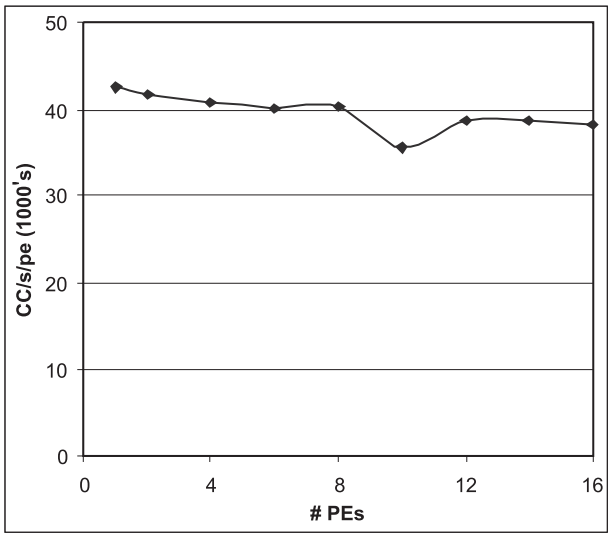

Fig. 16 Scaling behavior of SAGE on the EV7 machine (in node) using the CC/s/pe metric.

The performance degrades as the number of cells increases. This is expected due to the limited cache capacity. On the smaller problem sizes, a large utilization of the L2 cache is possible. On the larger problem sizes, very little re-use of L2 cache is possible, resulting in a large utilization of main memory.

The performance levels off above 125,000 cells, when main memory is mainly utilized. The dips in performance are due to the number of cells being close to or an exact power of 2 (such as 4096 and 32,768). Having such a number of cells results in poor cache performance due to ping-pong interference causing a higher degree of cache misses.

The performance decrease over the range of spatial grid sizes considered is only 40%. This is a comparatively small decrease and is attributed to the good main memory bandwidth of the EV7 processor.

A comparison of the performance of a 1.2 GHz EV7 processor to a 1.25 GHz EV68 processor (as used in the current AlphaServer ES45 nodes) is shown in Figure 15. As can be seen in that figure, the performance on the EV7 is over 20% better than that achieved on the EV68 of a similar clock speed for larger problem sizes. This again reflects the high main memory bandwidth of the EV7. However, on smaller problem sizes, the EV68 achieves a better performance by up to 15% due to its larger L2 cache – 16 MB versus 1.75 MB for the EV7.

(ii) SAGE – Scalability. A scalability test of SAGE was performed on between 1 and 16 processors contained within the EV7 node. The number of cells per processor was set at a constant of 13,500 and thus forms a weak-scaling study.

Results are shown in Figure 16 using the number of cells processed in one second per processing element (cc/s/pe) metric. The cc/s/pe should ideally be constant but decreases slightly due to parallel overhead.

It can be seen from Figure 16 that the performance decrease up to 16 processors is small (from 42,500 on a single processor down to 38,200 on 16 processors). This represents a high efficiency of 90% on 16 processors. The unexpected performance on 10 processors can most likely be attributed to a poorer cache utilization that can occur from the only approximately weak-scaling behavior of SAGE.

4.2 MCNP

The performance of MCNP is examined here in a strong scaling study. The number of particles that are processed in each cycle was set at a constant and divided up across the number of slaves available for processing. The geometry in which the particles move was replicated across the processors being used. The processing of each particle within a cycle is independent and thus communication

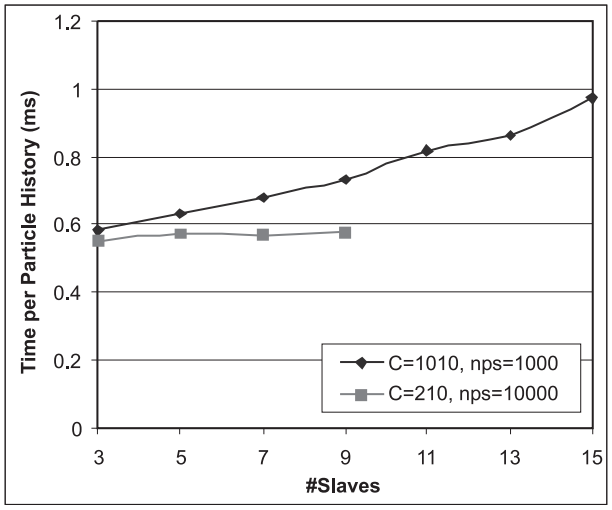

Fig. 17 Time per particle history in MCNP using the critexp input deck on the EV7 node.

occurs at the start and end of a cycle between all slaves and a master processor. The number of cycles, *C*, and the particle histories per cycle, *nps*, were set to be (*C* = 1010, *nps* = 1,000) and (*C* = 210, *nps* = 10,000) in two separate scalability tests.

The time per particle history while varying the number of slave processors used is shown in Figure 17. Note that the number of processors used in each case is actually the number of slaves + 1 (i.e., plus a master processor that accumulates results). The code is subject to a high degree of communication on the smaller problem (*C* = 1010, *nps* = 1,000) resulting in a low efficiency when using all 15 slaves (<60%).

The scalability of the larger problem (*C* = 210, *nps* = 10,000) is expected to be much better due to a decrease in the degree of communication. The time per particle history when using three slave processors on the larger problem is ~0.55 ms on the EV7 node. Not all configurations of the larger problem size were run, but the few measurements made indicate a better scaling behavior than on the smaller problem.

4.3 SWEEP3D

The performance of Sweep3D was also measured on the EV7 node. The performance of a 50-cubed problem size with 1-k plane per block and one angle per block was examined. The total run time was measured in a weak-scaling analysis. Observed parallel efficiency on 16 processors was about 90% on the EV7 node. The single-processor time on EV7 was also found to be 30% faster than the single-processor time on an EV68 (1 GHz) ES45 processor.

5 Comparative Performance

The results of the performance tests measured on the EV7 machine are compared with the performance obtained on current AlphaServer systems in this section. The comparisons are made on a like-for-like basis unless stated. A comparison of performance is shown in Table 2. No inter-node MPI performance was measured for an EV7 system as we had access to only a single node. Also included for comparison in Table 2 is the achieved performance on SAGE in a number of configurations.

Recall that the EV7 node that was benchmarked is a prototype and that the measured performance may not reflect the achieved level of performance possible on production systems after system and application tuning.

We note the following about Table 2:

(1) Remote memory latency on the EV7 varies on the distance (PE hops) between data locality and PE accessing data.
(2) The memory bandwidth on the ES40 and ES45 decreases depending on the number of PEs simultaneously accessing memory (a decrease up to a factor of 2 is possible). No decrease occurs on the EV7.
(3) Peak values for inter-node latency and bandwidths are quoted. These can decrease depending on distance between nodes, and physical lengths of wires used.

The performance comparisons show a number of significant performance improvements of the EV7 1.2 GHz machine relative to the existing EV68 ES45 1.25 GHz machine. These can be summarized as follows:

- the main memory bandwidth is a factor of 2 better;
- intra-node MPI latency is almost a factor of 3 better;
- intra-node MPI bandwidth is 30% better.

6 Summary

The performance evaluation of the new generation AlphaServer has shown that the EV7 processor has excellent main memory bandwidth, which is almost a factor of 2 greater than that of existing Alpha processors. In addition, there is only a factor of 2 decrease in the memory read bandwidth between L1 cache (7.77 GB/s) and main memory (4.6 GB/s). The bandwidth from remote memory within the node is also high at approximately 3.6 GB/s. The small L2 cache (1.75 MB) has a negative impact on application performance on larger problem sizes.

Table 2
Comparison of various performance characteristics of Alpha machines.

	ES40 (EV68)	ES45 (EV68)	ES45 (EV68)	EV7
System Characteristics				
Clock speed	833 MHz	1 GHz	1.25 GHz	1.2 GHz
Node size (CPUs)	4	4	4	16
L1 Cache	64 KB	64 KB	64 KB	64 KB
L2 Cache	8 MB	8 MB	16 MB	1.75 MB
Memory Performance				
Latency (cycles): L1	2	2	2	2
L2	12	19	19	12
Main	168	170	170	106
Remote Memory	–	–	–	162–290[1]
Read Bandwidth (GB/s): L1	4.93	6.47	7.89	7.77
L2	3.97[2]	6.07[2]	7.52[2]	6.20
Main	1.70[2]	2.27[2]	2.27[2]	4.58
Remote Memory	–	–	–	3.60
MPI performance				
Intra-Node (Point to Point)				
Unidirectional: Latency (μs)	6.2	4.9	4.9	1.7
Bandwidth (MB/s)	695	792	792	1,080
Bidirectional: Latency (μs)	12.7	8.9	8.9	2.2
Bandwidth (MB/s)	317	379	379	485
Inter-Node[3] (QsNet – Elan3)				
Unidirectional: Latency (μs)	5.6	4.5	4.5	–
Bandwidth (MB/s)	199	293	293	–
Bidirectional: Latency (μs)	9.8	7.4	7.4	–
Bandwidth (MB/s)	79	132	132	–
Application Performance				
Sage 13,500 cells: 1PE	30,500	38,300	49,400	42,500
16PEs	23,200	27,900	36,100	38,200
195,112 cells: 1PE	16,200	23,000	27,000	32,600

The MPI communication performance compares well with current systems. Point-to-point message latency between adjacent processors is low at 1.7 μs and bandwidth between adjacent processors is just over 1 GB/s. However, the latency is high when compared to remote memory latency (1.7 μs versus 135–240 ns), and the bandwidth is low when compared to peak remote memory bandwidth (1 GB/s versus 3.6 GB/s).

The MPI latency increases as the distance between processors increases, with the maximum being 2.4 μs. The bandwidth between any two processors is a constant at just over 1 GB/s. However, the barrier latency was 11.2 μs for all 16 processors. This seems large when compared with clusters interconnected with Quadrics's QsNet (Petrini et al., 2002). The broadcast bandwidth also decreases as the number of processors increases due to a lack of hardware support; the bandwidth for all 16 processors was only 300 MB/s.

The application performance showed that in-node scaling was good, resulting in high efficiencies on most codes (90% at 16 processors for SAGE, and SWEEP). On a detailed analysis of scaling the spatial grid in SAGE, the performance decrease from running a small problem (cache bound) to a large problem (main memory bound) was only 24%.

Due to the excellent main-memory bus bandwidth, a higher performance should be achievable on nodes using the EV7 processor in comparison to the existing EV68 at a similar clock-speed used in the existing AlphaServer ES45 nodes.

ACKNOWLEDGMENTS

The authors wish to thank Richard Foster, Niraj Srivastava, Zarka Cvetanovic, and Ed Benson for access to the EV7 node and technical discussions on the performance of the AlphaServer systems.

BIOGRAPHIES

Darren Kerbyson is currently a member of the technical staff in the Performance and Architectures Laboratory part of the Modeling, Algorithms and Informatics group (CCS-3) at LANL. He received his B.Sc. in Computer Systems Engineering in 1988, and Ph.D. in Computer Science in 1993 both from the University of Warwick (UK). Before his appointment at LANL he was a Senior Lecturer in Computer Science at the University of Warwick. His research interests include performance evaluation, performance modeling, and performance optimization of applications on high performance systems as well as image analysis. He has published over 60 papers in these areas over the last 10 years. He is a member of the Association for Computing Machinery, and the IEEE Computer Society.

Adolfy Hoisie is the group leader of the Modeling, Algorithms and Informatics Group in the Computer and Computational Sciences Division at LANL. From 1987 until he joined LANL in 1997, he was a researcher at Cornell University. His area of research is performance evaluation of high-performance architectures. He has published extensively, lectured at numerous conferences and workshops, often as an invited speaker, and taught tutorials in this field at important events worldwide. He was the winner of the Gordon Bell Award in 1996, and co-author to the recently published SIAM monograph on performance optimization.

Scott Pakin is a member of the technical staff in the Performance and Architectures Laboratory, part of the Modeling, Algorithms and Informatics group (CCS-3) at LANL. His current research interests include analyzing and improving the performance of high-performance computing systems, with particular emphasis on the memory and communication subsystems. He has published papers on topics such as high-speed messaging layers, job-scheduling algorithms, and resource-management systems. He received a B.S. in Mathematics/Computer Science with Research Honors from Carnegie Mellon University in 1992, an M.S. in Computer Science from the University of Illinois at Urbana-Champaign in 1995, and a Ph.D. from the University of Illinois at Urbana-Champaign in 2001.

Fabrizio Petrini is a member of the technical staff in the CCS-3 group at LANL. He received his Ph.D. in Computer Science from the University of Pisa in 1997. Before his appointment at LANL he was a research fellow in the Computing Laboratory at Oxford University (UK), a postdoctoral researcher of the University of California at Berkeley and a member of the technical staff at the Hewlett Packard Laboratories. His research interests include performance evaluation, job-scheduling algorithms, high-performance interconnection networks and network interfaces, simulation of parallel architectures, operating systems and parallel programming languages. He was the recipient of the European Community Marie Curie Fellowship in 1997, the Los Alamos Performance Award and the National Nuclear Security Agency (NNSA) Excellence Award, both in 2002.

Harvey Wasserman has been a Staff Scientist at LANL since 1985. His research interests involve supercomputer architecture and performance evaluation, and he has participated in benchmarks of almost all significant high-performance computing architectures, including single-processor workstations, parallel vector supercomputers, and massively-parallel systems. In a prior life he was a chemist and he holds a Ph.D. in Inorganic Chemistry from the State University of New York and was a Postdoctoral Research Associate at Los Alamos in 1982–1984. In 1999 during a one-year sabbatical at LANL he developed and taught a curriculum on ASCI system usage. He is a co-author of over 50 articles and has presented numerous invited and contributed lectures and tutorials.

REFERENCES

Compaq, 2002. Compaq AlphaServer ES45 Systems – Technical Summary, available at http://www.compaq.com/AlphaServer.

Cvetanovic, Z. and Kessler, R.E., June 2000. Performance analysis of the Alpha 21264-based Compaq ES40 system. *Proceedings of the 27th ISCA*, Vancouver, Canada, pp. 192–202.

Galles, M., 1987. Spider: A high-speed network interconnect. *IEEE Micro* 17(1):34–49.

Kerbyson, D.J., Alme H.J., Hoisie A., Petrini F., Wasserman H.J., Grillings M., 2001. Predictive Performances and Scalability Modeling of a large-scale Application. Proceedings SC'01, Denver November 10-16 2001.

Koch, K.R., Baker, R.S., and Alcouffe, R.E., 1992. A Parallel Algorithm for 3D Sn Transport Sweeps, LA-CP-92-406, LANL.

Krewell, K., April 2002. Alpha EV7 Processor: A High Performance Tradition Continues, Microprocessor Report, San Jose, CA.

McKinney, G., 1994. A practical guide to using MCNP with PVM. Transactions of the American Nuclear Society 71:397.

Mucci, P. and London, K., 1998. Low Level Architectural Characterization Benchmarks for Parallel Computers, Technical Report UT-CS-98-394, University of Tennessee.

Mukherjee, S.S., Bannon, P., Lang, S., Spink, A., and Webb, D., 2002. The Alpha 21364 network architecture. *IEEE Micro* 22(1):26–35.

Petrini, F., Feng, W.C., Hoisie, A., Coll, S., and Frachtenberg, E., 2002. The Quadrics network: high-performance clustering technology. *IEEE Micro* 22(1):46–57.